מבוא לבינה מלאכותית - 236501

תרגיל בית 1

מרחבי חיפוש

הוגש ע"י: מוחמד גנאים

מטרות התרגיל

- נתמודד עם בעיות פרקטיות ותיאורטיות של חיפוש במרחבי מצבים.
- נתרגל את הנלמד בהרצאות ובתרגולים.
- נתנסה בתכנות ב-python לפתרון בעיות פרקטיות.

הנחיות כלליות

- תאריך הגשה:**29.2** יום חמישי, בשעה 23:59.(לשנות מועד ולפתוח פיאצה)
- את המטלה יש להגיש בזוגות בלבד.
- יש להגיש מטלות מוקלדות בלבד בעברית או באנגלית. פתרונות בכתב יד לא ייבדקו.
- ניתן לשלוח שאלות בנוגע לתרגיל בפיאצה בלבד.
- המתרגל האחראית על תרגיל: **שאדי דיאב** .
- בקשות דחיה מוצדקות (מילואים, אשפוז וכו') יש לשלוח למתרגל האחראי (**ספיר טובול**) בלבד.
- במהלך התרגיל ייתכן שנעלה עדכונים, למסמך הנ"ל – תפורסם הודעה בהתאם.
- העדכונים הינם מחייבים, ועליכם להתעדכן עד מועד הגשת התרגיל.
- שימו לב, התרגיל מהווה כ- 15% מהציון הסופי במקצוע ולכן העתקות תטופלנה בחומרה!
- ציון המטלה יורכב מהגורמים הבאים:
  - 65% - המסמך היבש.
  - 35% - הקוד המוגש.
- אנו יודעים שעבור חלקכם זו התנסות ראשונה בכתיבת קוד בפיתון ותרגיל זה מתוכנן בהתאם לכך.
- שימו לב שלא יענו שאלות בסגנון: "איך מוצאים את עלות הפתרון שהוחזר?" / "איך ניגשים למפות הכבישים מתוך המימוש של הפונק' ההיא?" / "באיזה שדה שמור ה...?" וכדומה.
- אנחנו רוצים לעודד אתכם לעיין בקוד ולמצוא פרטים אלו בכוחות עצמכם. הכרת סביבת העבודה שסיפקנו לכם והתמצאות בה הן למעשה חלק מהתרגיל.
- בתרגילי הבית בקורס הרצת הניסויים עשויה לקחת זמן רב. לכן מומלץ מאוד להימנע מדחיית העבודה על התרגיל ו/או כתיבת הדו"ח לרגע האחרון. לא תינתנה דחיות על רקע זה.
- מסמך זה כתוב בלשון זכר מטעמי נוחות בלבד, אך מתייחס לנשים וגברים כאחד.

אנחנו קשובים לפניות שלכם במהלך התרגיל ומעדכנים את המסמך הזה בהתאם. גרסאות עדכניות של המסמך יועלו לאתר. הבהרות ועדכונים שנוספים אחרי הפרסום הראשוני יסומנו כאן בצהוב. בנוסף, לכל עדכון יהיה מספר גרסה כדי שתוכלו לעקוב. ייתכן שתפורסמנה גרסאות רבות – אל תיבהלו מכך. השינויים בכל גרסה יכולים להיות קטנים.

הנחיות לחלק היבש

1. ככלל אצבע, בהינתן שאלה ראשית ספקו את התשובה המיידית ולאחר מכן תרחיבו ותסבירו. למשל, אם שואלים מה סיבוכיות הזמן של אלגוריתם BFS תשובה תהיה "$O(b^d)$, מכיוון שבקרה הכי גרוע נאחסן את כל עץ החיפוש של הבעיה בCLOSE".

הנחיות לחלק הרטוב

1. אנו מעודדים אתכם לעבור על הקבצים המצורפים ולהבין כיצד הסביבה בנויה ובאילו פונקציות תוכלו להשתמש במימוש שלכם.

2. הקוד שלכם ייבדק בקפדנות על ידי טסטים. הטסטים יבדקו את הפתרונות המוחזרים על ידי האלגוריתמים שלכם אל מול המימוש שלנו על פני בעיות שונות. אנו מצפים ממכם (אלא אם צוין אחרת) להחזיר את אותם ערכים בדיוק. אנחנו נבדוק את המסלול המוחזר, מספר הצמתים שפתחו ואת עלות הפתרון המוחזר. הטסטים יהיו מוגבלים בזמן אך תקבלו זמן גדול מאוד לכל טסט.
3. ספקו קוד ברור ונקי הניתן לבדיקה ידנית.

מבוא ורקע

התרגיל מתפרש על פני <u>מסמך זה והמחברת המצורפת</u>. מומלץ לענות על השאלות לפי הסדר במסמך זה.

במטלה זו נעסוק בהפעלת אלגוריתמי חיפוש על מרחבי מצבים לבעיות ניווט. מומלץ לחזור על שקפי ההרצאות והתרגולים הרלוונטיים לפני תחילת העבודה על התרגיל.

סיפור מסגרת

לקאקרוטו וגוהאן יש 5 כדורי דרקון וחסר להם שני כדורים, והם ממש צריכים אותם כדי להזמין הדרקון שן-ראן ולבקש ממנו להחזיר את החברים שלהם לחיים, לכן הם הלכו לכוכב לכת נאמיק כדי לחפש כדורי הדרקון, קאקרוטו הציע שיחפשו על הכדור דרך ה ג׳׳פי.אס שלהם אבל גוהאן מסביר לקאקרוטו שיש לו חברים שלוקחים הסמסטר את קורס ״מבוא לבינה מלאכותית״. גוהאן מבקש ממכם לעזור לו לתכנן את המסלול הטוב ביותר כדי לאסוף כדורי הדרקון ולהגיע לקאקרוטו שמחקה לו.

שאלה 1 – מבוא (8 נק׳):

השאלות בחלק זה מתבססות על הלוח "8x8" שמופיע במחברת אלא אם נכתב אחרת:

1. **רטוב**: עברו על המחברת עד שאתם מגיעים לחלק של BFS-G ועצרו שם.
2. יבש (1 נק׳): תחילה נרצה להגדיר את מרחב החיפוש כפי שנלמד בתרגול. הגדר את (S, O, I, G) עבור סביבת כדורי הדרקון. כאשר S זה מרחב המצבים, O, זה מרחב האופרטורים, I, זה המצב ההתחלתי וG הוא קבוצת מצבי המטרה. מה גודל מרחב המצבים S? הסבירו.

<u>The search problem environment for the DragonBall problem is {S,O,I,G} where S,O,I,G are the following:</u>

- <u>S: each state 's' is described as the following:</u>

s = (Place in Board, DragonBall1 Collected, DragonBall2 Collected)
Place in Board = {0,1,2,….,62,63}.
DragonBall1 Collected = DragonBall2 Collected = True/False or {0,1}
|S| = 64*2*2 = 256.

- O: The set of operators are O = {DOWN,RIGHT,UP,LEFT}
- I: The initial state in this specific problem is (0,False,False) where 1 is the top left square in the board and none of the 2 DragonBalls are collected.
- G: (63,True,True). The goal of this specific problem is to reach the destination square, which in this case is the 64th square in the bottom right corner of the board, and we should reach that square when both DragonBalls are collected.

3. יבש (1 נק׳): מה תחזיר לנו הפונקציה Domain על אופרטור 2 (UP)?
$Domain(UP) = \{s \in S \mid UP(s) \neq \emptyset\} = \{8,9, \dots ,62,63\}.$

4. יבש (1 נק׳): מה תחזיר לנו הפונקציה Succ על המצב ההתחלתי 0?
Succ(0) will return the 3 possible states, where each operator out of the 4 will return a state of its own which describes the square we move to after applying the operator, the cost of applying it and moving to that square, and whether the game is Terminated or not. Since both operators UP/LEFT return the same output we consider 3 possible states and not 4.

5. יבש (1 נק׳): האם קיימים מעגלים במרחב החיפוש שלנו?
Yes. For example we can move from 0 → 1 → 9 → 8 → 0.

6. יבש (1 נק׳): מה הוא מקדם הסיעוף בבעיה?
Our branching factor is 4.

7. יבש (1 נק׳): במקרה הגרוע ביותר, כמה פעולות ידרשו לסוכן כללי להגיע למצב הסופי?
The number of operations is unbounded since we are not promised we will ever reach the goal state.

8. יבש (1 נק׳): במקרה הטוב ביותר, כמה פעולות ידרשו לסוכן כללי להגיע למצב הסופי?
16 steps. The fastest way to collect both DragonBalls in the 48th and the 53rd squares is to first collect the one in the 48th square, then the one in the 53rd square. After that we can finally move to the 63rd square which is our Goal. The fastest route from square 0 to square 48 takes 6 steps. And the fastest route from the 48th square to the 53rd one takes 7 steps. And finally from the 53rd square to the Goal square we need 3 steps. Which totals to 16 step.

9. יבש (1 נק׳): עבור לוח כללי, המסלול הקל ביותר הוא המסלול שמגיע למצב מטרה שהכי קרוב למצב ההתחלתי (במונחים של manhattan distance)? אם כן, הוכיחו. אם לא, ספקו דוגמא נגדית.
True. Since a 'G' square costs 1 and there is no other type of square that costs less. Lets assume that we found a path lighter than a path from the initial square to the goal square which is in a neighboring square. The path from the initial square to the 'Goal' square costs 1. Meaning that the lightest path we found costs less than 1, meaning that we had to go through a square that costs less than 1. Since the minimum cost of a square is 1, the assumption is false.

שאלה 2 – Breadth First Search-G (7 נק׳):

השאלות בחלק זה מתבססות על הלוח "8x8" שמופיע במחברת אלא אם נכתב אחרת.

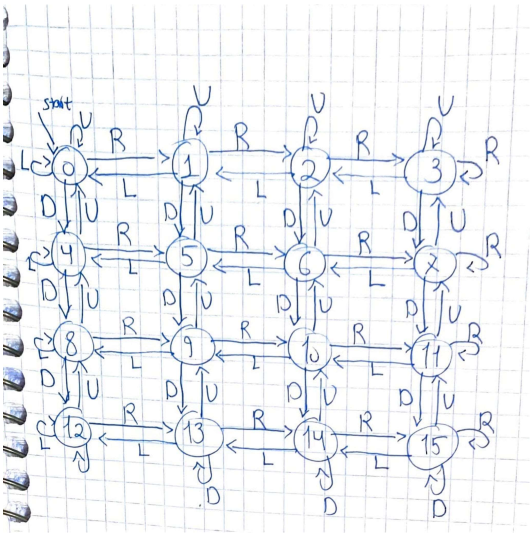
1. **רטוב**: ממשו את אלג׳ BFS-G (על גרף) במחברת ע״פ ההנחיות המופיעות שם.

2. יבש (1 נק׳): מה צריך להיות התנאי על גרף החיפוש (לא בהכרח בבעיית כדורי הדרקון) כך שBFS על גרף ו-BFS על עץ ייצרו ויפתחו צמתים זהים באותו הסדר?

The BFS-Graph must be a BFS-Tree in order to to expand and create vertices in the BFS-G in the same way as in BFS-T. The reason behind that is that BFS-T expands and creates vertices based on its state

Tree and it doesn't visit a vertice twice, and BFS-G does the same based on its state Graph but since it's a graph that has circles in it it can visit a vertex more than once. Therefor the graph should have no circles which makes it a Tree.

3. יבש (2 נק׳): עבור הלוח "4x4" שמופיע במחברת, ציירו את גרף המצבים.



4. יבש (2 נק׳): נתון לוח בגודל NxN. הציעו דרך להשתמש באלגוריתם BFS-G כך שיחזיר פתרון אופטימלי (עלות מינימלית) והסבירו.
   - רמז: עליכם לספק פונקציה $T: G \to G'$ המקבלת את גרף המצבים G ויוצרת גרף חדש G' ובעזרתה למצוא את המסלול האופטימלי בגרף G.

We define a function T:G→G' as the following:

- Given a state-space graph G where each transition from state i to state j through an edge has its own added cost, replace the edge connecting between state i and state j with |cost(i,j)| vertices and connect between each of those vertices with an edge of weight = 1.
- Return the new graph G'.

The returned graph G' replaced every path from state i to state j with a similar path that goes through |cost(i,j)| -1 vertices. but now all the edges in the new graph G' are of weight 1. Meaning that we can run BFS on G' and get the minimal cost of reaching the goal state from the initial state which is now the distance from those 2 states in G'.

In order to get the path in the original graph G we reverse what we did and replace the added vertices with the original edge between the 2 states.

5. יבש (2 נק׳): נתון לוח בגודל NxN, ללא חורים, המכיל $N^2 - 2$ משבצות רגילות (F,T,A,L) מצב התחלתי בפינה השמאלית עליונה ומצב מטרה בפינה הימנית תחתונה. כמה צמתים יפותחו וייווצרו במהלך חיפוש BFS-G? הסבירו?

$Created: N^2$, $Expanded: N^2 - 1.$

The way BFS-G works is that it expands or creates vertices by their Manhattan distance from the initial vertex. In a NxN grid with no holes with the initial vertex being the one in the top left corner and the goal vertex being the one in the bottom right corner, the goal vertex is the farthest from the initial vertex in terms of its Manhattan distance. Therefor all the the vertices including the goal vertex will be created and all of them vertices except the goal vertex will be expanded.

שאלה 3 – Depth First Search-G (6 נק׳):

1. יבש (1 נק׳): עבור בעיית כדורי הדרקון עם לוח NxN, האם האלגוריתם שלם? האם הוא קביל?

The DFS-G algorithm **is not complete** since given a circle in the graph the algorithm can get stuck in a loop and never find the solution. Given that we would've added some mechanism to the implementation of DFS-G that prevents it from visiting nodes we already visited like in the case of BFS from the previous question then it would've been complete.
The algorithm is not complete, therefor it **can't be admissible by definition**.

2. יבש (1 נק׳): האם אלגוריתם DFS (על עץ), עבור בעיית כדורי הדרקון על לוח NxN, היה מוצא פתרון כלשהו? אם כן, מה המסלול שיתקבל? אם לא, כיצד האלגוריתם היה פועל?

Yes. In this case the algorithm can never be stuck in an infinite loop since there are no circles. The tree is finite therefor we are promised to find a solution. The returned path would depend on how deep the vertex we are looking for is and where do we start from. And it will include all the ancestors of the goal vertex starting from the initial vertex, and maybe some other vertices that are not ancestors of the goal vertex in case the algorithm backtracks at some point.

3. יבש (2 נק׳): נתון לוח בגודל NxN, ללא חורים, המכיל N^2-2 משבצות רגילות (F,T,A,L) מצב התחלתי בפינה השמאלית עליונה ומצב מטרה בפינה הימנית תחתונה (תניחו כי שני כדורי הדרקון הם בפינה ימינית תחתונה) . כמה צמתים יפותחו וייווצרו במהלך חיפוש DFS-G? הסבירו?

$Created: 4N-5\ , Expanded: 2N-2.$

In our specific definition of the DragonBallEnv, the priority of the actions is {DOWN,RIGHT,UP,LEFT}. Therefor starting from the top left corner vertex we go DOWN N-1 times until we reach the (N,1) vertex/square in the board and right after we keep going RIGHT N-1 times starting from the (N,1) square up until the (N,N) square which is our goal square. Excluding the goal vertex which we do not expand that is exactly 2N-1-1=2N-2 expanded vertices.
The created vertices are all the vertices in the 1st and 2nd column, as well as all the vertices in the nth and n-1th row except the vertex above our goal vertex since we reach the (N,N) goal vertex and we do not expand it.

4. יבש (2 נק׳): נתון לוח בגודל NxN, ללא חורים, המכיל N^2-2 משבצות רגילות (F,T,A,L) מצב התחלתי בפינה השמאלית עליונה ומצב מטרה בפינה הימנית תחתונה (תניחו כי שני כדורי הדרקון הם בפינה ימינית תחתונה). כמה צמתים יפותחו וייווצרו במהלך חיפוש DFS-G backtracking? הסבירו?

$Created: 2N-1\ , Expanded: 2N-2.$

The difference between DFS-G and DGS-G backtracking is in the way it creates nodes. BFS-G backtracking creates nodes in a lazy way. Meaning that along the path DOWN from the start until it hits the bottom row it only creates the next vertex its about to expand in each step. The same for its path RIGHT along the last row until we reach our goal vertex. Therefor it creates all the vertices in the 1st column and Nth row(including the goal vertex).

שאלה 4 – ID-DFS (6 נק׳):

1.

a. (1 נק׳) האם האלגוריתם שלם? אם כן, הוכיחו. אם לא, ספקו דוגמה נגדית.

The algorithm is indeed complete given that the graph is finite. Lets assume a goal vertex of depth K>0 in the graph. The way the algorithm works is exactly like the DFS-L algorithm but with an increasing L each time we can't visit anymore vertices. meaning that the algorithm will visit all the vertices in the current Lth depth in the tree and then all the vertices in the L+1th depth and so on until reaching L=K depth. Since our goal vertex is in the current L=K depth the algorithm will certainly find it.

b. (1 נק׳) נניח כי עלות כל פעולה היא 1, האם האלגוריתם קביל? אם כן, הוכיחו. אם לא, הסבירו.

Yes. Assume the goal vertex is with L depth. The algorithm will certainly not find a path to the goal vertex in the first L-1 runs of the algorithm since we do not visit vertices with L depth in those runs. Now consider we are in the Lth run of the algorithm. We are promised to find a solution during that run that has exactly L edges of weight=1 each.

Assume we found a less costly path in a K>L run of the algorithm. This path has atleast K edges of weight=1. meaning that the path costs exactly K>L in contrary with our assumption and the path of L cost we found in the Lth run.

c. (1 נק׳) הציעו דרך לעדכן את אלגוריתם על מנת לתקן את הבעיה מהסעיף הקודם.

?

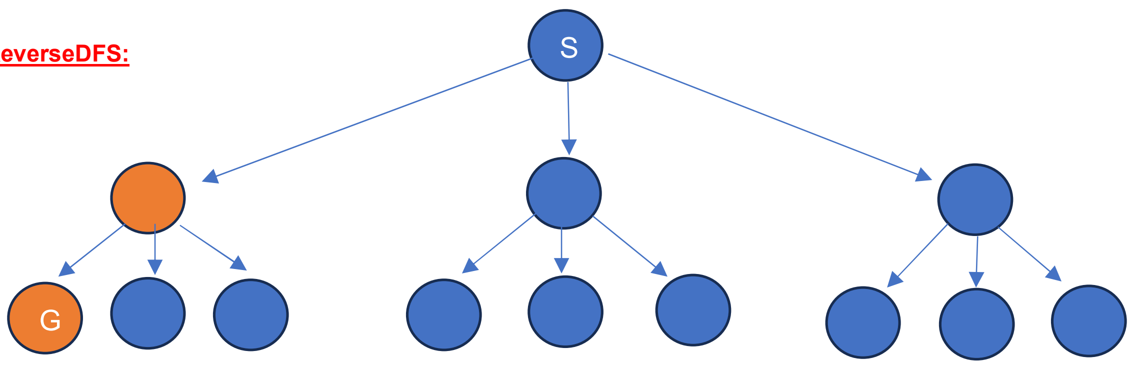
2. הניחו כי יש לנו ידע מקדים על חסם עליון למרחק למצב מטרה, נסמן D. בת (Beth) הציעה את האלגוריתם חיפוש הבא:

```
function ReverseDFS (problem, D):

        L ← D
        result ← failure
        While Not Interrupted:
                new_result ← DFS-L (problem, L)
                if new_result = failure:
                        break
                L ← L - 1
                result ← new_result

        return result
```

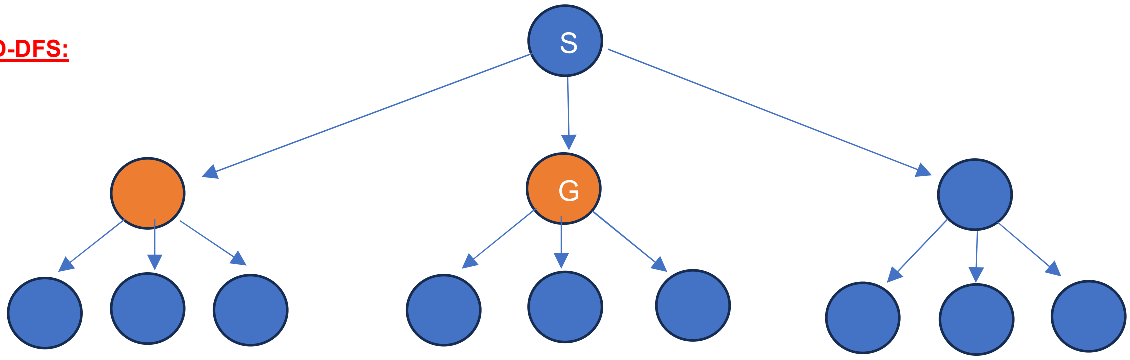
3. בשאלות הבאות הניחו כי יש מספיק זמן לסיום האיטרציה הראשונה.

a. (1 נק׳) ספקו דוגמה בה ReverseDFS עדיף על ID-DFS ודוגמה בה ID-DFS עדיף על ReverseDFS. הדוגמאות יכולות להיות כלליות ולא בהכרח מסביבת התרגיל.

ReverseDFS:



ID-DFS:



In the 1st graph, ReverseDFS is better since the goal vertex is in the bottom left vertex with the maximal depth. If we ran ID-DFS in that graph we would've visited all the vertices with depth=1 before running the algorithm again with L=2 and finding our solution.

In the 2nd graph, ID-DFS is better since our goal vertex is in depth=1. If we ran ReverseDFS on that graph, we would've visited all the vertices with depth=2 in the left corner before backtracking to the root and visiting our goal vertex.

b. (2 נק׳) הציעו כיצד ניתן לייעל את האלגוריתם. רמז: האם אתם יכולים לחשוב על צעד עדכון עדיף לL?
נשנה את האלגוריתם בצורה הבאה :

Instead of starting the algorithm with L=D and decrementing with 1 upon failure, lets consider the following implementation:

Binary Search on Depth:

- Calculate the midpoint midDepth = (minDepth + maxDepth) / 2. Where minDepth=0 and MaxDepth=D.
- Run DFS-L at midDepth.
- If DFS-L succeeds, the goal is at midDepth or shallower. Set maxDepth = midDepth and repeat the search in the lower half.
- If DFS-L fails, the goal must be deeper. Set minDepth = midDepth + 1 and repeat the search in the upper half.
- Continue this process until minDepth and maxDepth converge.

שאלה 6 - UCS (4 נק׳):

השאלות בחלק זה מתבססות על הלוח "8x8" שמופיע במחברת אלא אם נכתב אחרת.

1. יבש (1 נק׳): עבור אילו בעיות חיפוש אלגוריתם UCS ואלגוריתם BFS יפעלו באותו האופן? הסבירו.

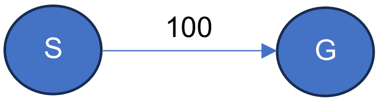
In graphs with equal weights on all edges since in that case the optimal solution is also the shortest one.

2. יבש (1 נק׳): האם בבעיית החיפוש שלנו, עבור לוח NxN, האלגוריתם הוא שלם? האם הוא קביל?

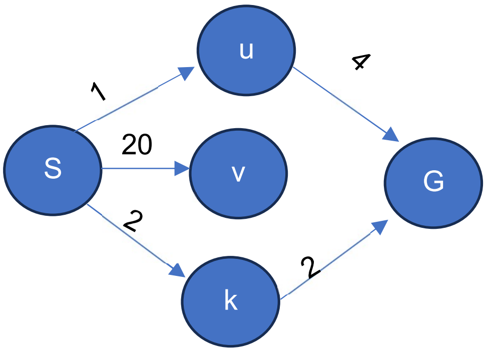
As shown in class, the UCS algorithm is both complete and admissible.

3. יבש (2 נק'): שאדי טעה במימוש של אלגוריתם UCS ובטעות בדק בעת יצירת הצומת האם היא צומת מטרה במקום בפיתוח שלה. הביאו דוגמה לגרף חיפוש שעבורו שאדי יחזיר בכל זאת את המסלול הקל ביותר, ודוגמה לגרף חיפוש שעבורו שאדי לא יחזיר את המסלול הקל ביותר. עבור כל דוגמה הסבירו מה המסלול והעלות ש-UCS השגוי החזיר, ומה המסלול והעלות שהאלגוריתם הנכון היה מחזיר. נדגיש שגרף החיפוש לא בהכרח צריך לייצג את בעיית כדור הדרקון. אתם יכולים לתת דוגמה לגרף שמייצג בעיית חיפוש אחרת. הגרף צריך להכיל קשתות מכוונות ואת העלות של כל קשת.

1st graph:



2nd graph:



The first graph has only 1 edge and it's the one from the start to the goal vertex so both the good and bad UCS return the same path with cost=100.

In the second graph, the bad UCS first chooses the (s,u) edge and then the (u,G) edge and returns a path with cost=5. The good UCS chooses the (s,u) as well in the beginning, then the (s,k) edge and the (k,G) edge and returns a path with cost=4 which is the optimal path.

שאלה 7 - יוריסטיקות (8 נקי'):

יהי מרחב חיפוש (S,O,I,G) , נסתכל על בעיית הניווט לכדור דרקון יחיד. המטרה היא למצוא מסלול זול ביותר מהמוצא I ליעד יחיד G . פונק' העלות מוגדרת כאורך הכביש המחבר בין שתי נקודות, יהיה חסם תחתון $\delta > 0$ על אורך הכבישים. ניתן להניח כי העולם שטוח . מלבד זאת, לא ניתן להניח דבר נוסף על מרחב החיפוש.

הגדרה : יוריסטיקה h היא ε-קבילה אם קיים $\varepsilon \geq 1$ כך שלכל מצב $s \in S$ מתקיים $h(s) \leq \varepsilon \times h^*(s)$.
נזכיר כי $h^*(s)$ הינה פונקציית המחיר המסלול האופטימאלי מ-s לצומת היעד .

עבור כל אחת מהיוריסטיקות הבאות קבעו האם קיים $\varepsilon \geq 1$ כך שהיוריסטיקה תהיה ε-קבילה . אם כן מצאו את ה-ε ההדוק ביותר המקיים את זאת. נמקו היטב .

1. יבש (1 נקי'): מרחק מנהטן : $h_{MD}(p) = |P - G|_1 = |G_x - P_x| + |G_y - P_y|$

Yes it is, $\epsilon_{MIN} = 1$.
$h_{MD}^{MAX}(s) = G_x + G_y \; for \; s = I(P_x = P_y = 0)$.
$h^*(s) \geq (G_x + G_y) * \delta$
⇨ $\frac{h^*(s)}{\delta} \geq (G_x + G_y) = h_{MD}^{MAX}(s) \geq h_{MD}(s)$
⇨ $for \; \delta < 1: let \; \epsilon = \frac{1}{\delta}. => \epsilon * h^*(s) = \frac{h^*(s)}{\delta} \geq h_{MD}^{MAX}(s) \geq h_{MD}(s)$
⇨ $for \; \delta \geq 1: let \; \epsilon = 1. => h^*(s) * \epsilon = h^*(s) \geq \frac{h^*(s)}{\delta} \geq h_{MD}^{MAX}(s) \geq h_{MD}(s)$.

2. יבש (1 נקי'): $h(p) = |P - G| = \min\{G_x - P_x \; , \; G_y - P_y\}$

Yes it is, $\epsilon_{MIN} = 1$.

$h(p)^{MAX} = G_x = G_x < h_{MD}^{MAX}(s) = G_x + G_y$
$therefor \; h(p) \leq h^{MAX}(p) = G_x \leq h^*(p) = 1 * h^*(p) = \epsilon * h^*(p) \; from \; previously$.

3. יבש (1 נקי'): L^3 : $h(p) = |P - G|_3 = \sqrt[3]{|G_x - P_x|^3 + |G_y - P_y|^3}$

Yes it is, $\epsilon_{MIN} = 1$.
$h^{MAX}(p) = \sqrt[3]{G_x^3 + G_y^3} = \sqrt[3]{2G_x^3} = \sqrt[3]{2} * G_x$.
$h^*(p) \geq (G_x + G_y) * \delta \geq G_x * \delta$.
$for \; \delta \geq \sqrt[3]{2}: \quad \epsilon * h^*(p) = 1 * h^*(p) \geq G_x * \delta \geq \sqrt[3]{2} * G_x \geq h(p)$.
$for \; \delta < \sqrt[3]{2}: from \; previously, we \; got \; the \; following \; result: \; G_x \leq h^*(p)$.
$by \; multiplying \; with \; \sqrt[3]{2}, we \; get \; the \; following:$
$\sqrt[3]{2} * G_x = h^{MAX}(p) \leq \sqrt[3]{2} * h^*(p) = \epsilon * h^*(p) \; for \; \epsilon = \sqrt[3]{2}$.
$therefor, h(p) \leq \epsilon * h^*(p)$.

4. יבש (1 נקי'): נתונות יוריסטיקות h_1 , h_2 שהן $\varepsilon_1 , \varepsilon_2$ קבילות בהתאמה וכי $\varepsilon_1 , \varepsilon_2$ הם האפסילונים ההדוקים ביותר.
הראו כי $h_3 = h_1 + h_2$ היא ε_3 – קבילה , מצאו את ε_3 ההדוק ביותר והוכיחו .

$\epsilon_3 = \epsilon_1 + \epsilon_2$.
$h_1(p) \leq \epsilon_1 * h^*(p)$.
$h_2(p) \leq \epsilon_2 * h^*(p)$.

⇨ $h_3(p) = h_1(p) + h_2(p) \le \epsilon_1 * h^*(p) + \epsilon_2 * h^*(p) = (\epsilon_1 + \epsilon_2) * h^*(p) = \epsilon_3 * h^*(p).$

נגדיר יוריסטיקה חדשה :

- D היא קבוצת כדורי הדרקון , $D = \{d1, d2\}$.

$$h_{MSAP}(s) = \min\{h_{Manhatan}(s,g) | g \in G \cup D\}$$

הערה: בנוסחת המרחק מתייחסים למיקום של צומת.
שימו לב שבמקרה זה אנחנו לוקחים את המינימום על פני כל צמתי היעד.

5. יבש (1 נק׳): האם היוריסטיקה h_{MSAP} קבילה על כל לוח? אם כן הסבר, אם לא הבא קודמה נגדית.

Yes it is.
we proved that manhattan distance from G is indeed admissible in the first sub question.
h_{MSAP} is the minimal Manhattan distance between g and the 2 dragon balls from s, which in its max value is equal to the Manhattan distance from the target node g that we proved admissible. which means $h^{MSAP}(s) \le h^{MD}(s) < h^*(s)$ *from the first sub question.*

6. יבש (1 נק׳): האם היוריסטיקה h_{MSAP} עקבית על כל לוח? אם כן הסבר, אם לא הבא דוגמה נגדית.(לחשוב אם היא עקבית ולתקן בהתאם)

No its not. Lets take for example a 5x5 board where all the paths from adjacent squares cost $\delta = 0.5$.
Lets assume that the 2 dragon balls are located in (2,2) and (3,4). And lets assume s=(0,0) and g=(5,5).
$h^{MSAP}(s) = 4. h^{MSAP}(0,1) = h^{MSAP}(1,0) = 3.$ *where* $(1,0)$ *and* $(0,1)$ *are the 2 succesors of* $s.$
$h^{MSAP}(s) - h^{MSAP}(s') = 4 - 3 = 1$ *for both successors.*
$cost(s,s') = 0.5$
$therefor, h^{MSAP}(s) - h^{MSAP}(s') > cost(s,s').$

נגדיר יוריסטיקה חדשה :

- D היא קבוצת כדורי הדרקון , $D = \{d1, d2\}$.

$$h_{new}(s) = \max\{h_{Manhatan}(s,g) | g \in G \cup D\}$$

7. יבש (1 נק׳): האם היוריסטיקה h_{new} קבילה על כל לוח? אם כן הסבר, אם לא הבא קודמה נגדית.

Yes it is. We already proved that Manhattan distance is admissible, and h_{new} is simply some maximal Manhattan distance, which is also admissible similar to the previous sub question.

8. יבש (1 נק׳): האם היוריסטיקה h_{new} עקבית על כל לוח? אם כן הסבר, אם לא הבא דוגמה נגדית.

Same example from before, but this time $h_{new}(s) = 10$ *and* $h_{new}(s') = 9$ but the results are the same.

שאלה 8 – Greedy Best First Search (3 נק׳):

השאלות בחלק זה מתבססות על הלוח "8x8" שמופיע במחברת אלא אם נכתב אחרת.

1. יבש (1 נק׳): האם האלגוריתם שלם? האם הוא קביל?

The algorithm is indeed complete like we proved in class because board is finite. And its not admissible as we learned in class.

2. יבש (2 נק׳): תנו יתרון וחיסרון של אלגוריתם Greedy Best first Search לעומת Beam Search.

Pros: Greedy BFS expands less nodes than Beam Search.
Cons: Greedy BFS is not admissible. Beam Search is.

שאלה 9 – W-A* (2 נק׳):

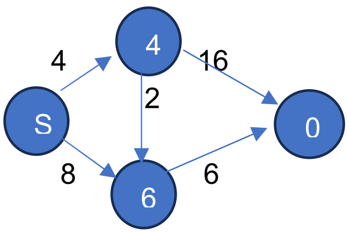
השאלות בחלק זה מתבססות על הלוח "8x8" שמופיע במחברת.

1. רטוב: ממשו את החלקים החסרים באלג׳ W-A* בקובץ ע״פ ההנחיות המופיעות שם. עליכם להשתמש ביוריסטיקה h_{MSAP}.

2. (יבש 2 נק׳) בהינתן $w_1 < w_2 \leq 1$, נסמן את המסלולים המחוזרים על ידי W-A* תחת הפורמולציה $f = g + w \cdot h$ ב p_1, p_2 עבור w_1, w_2 בהתאמה. אזי $cost(p_1) < cost(p_2)$ עבור:

   a. יוריסטיקה קבילה h. אם כן הסבירו. אם לא, ספקו דוגמה נגדית.

False



for w1=0.5: cost(p1)=12
for w2=1: cost(p2)=12
cost(p1)=cost(p2).

   b. יוריסטיקה כללית (לא בהכרח קבילה) h. אם כן הסבירו. אם לא, ספקו דוגמה נגדית.

Same graph from previously but this time h=10 for the goal state instead of 0. Which makes the heuristic not admissible. The 2 returned paths p1 and p2 and still the same.

שאלה 10 – IDA* (2 נק׳):

1. יבש (1 נק׳): ספקו יתרון וחסרון של IDA* ביחס לA*. באילו מקרים הייתם מעדיפים להשתמש בכל אחד מהם?

Pros: less memory usage.
Cons: might visit states that were already explored.

We would prefer to use IDA* in problems where there is a memory constraint. And we would prefer to use A* in problems where the search space is large.

2. יבש (1 נק׳): ספקו המחשה שלב אחר שלב של אלגוריתם IDA* על הלוח (4x4) שמופיע במחברת, המראה כיצד החיפוש מתקדם באמצעות העמקה איטרטיבית ?

It will start and explore the DOWN node and hit f_limit. Then it will explore the RIGHT node and restart with a new f_limit. It then will again go DOWN twice and hit the limit but before restarting it will explore the first Dragon Ball and other nodes. Then it will start again and hit the limit again but keeps exploring nodes with the current depth. Then it will go again and this time explore the second dragon ball and on the final run it will reach the goal node.

שאלה 11 – A* epsilon (6 נק׳):

1. **רטוב**: ממשו את החלקים החסרים באלג׳ W-A* בקובץ ע״פ ההנחיות המופיעות שם. עליכם להשתמש ביוריסטיקה h_{MSAP}.
2. יבש (2 נק׳): תנו יתרון וחיסרון של A*-epsilon לעומת A*.

One of the pros of A*-epsilon is that it might find a solution more quickly, but at the same time we might not return the optimal path or solution since finding it is based on our choice of the heuristic, and that is one of the cons of A*-epsilon in comparison with A*.

3. יבש (3 נק׳): תנו הצעה ליוריסטיקה כדי לבחור את הצומת הבאה לפיתוח מתוך FOCAL. תארו את היוריסטיקה והציגו השוואה בין השימוש ביוריסטיקה זו לעומת השימוש ב-$g(v)$, מבחינת מספר פיתוחים, מסלול שנבחר ועלות המסלול שנבחר.

We will choose the Euclidean distance like we saw in class:

When using g(v) with epsilon = 100:

```
Total_cost: 103.0
Expanded: 81
Actions: [0, 0, 0, 0, 0, 0, 1, 1, 1, 2, 1, 1, 0, 1, 1, 0]
```

when using Euclidean distance:

```
Total_cost: 103.0
Expanded: 81
Actions: [0, 0, 0, 0, 0, 0, 1, 1, 1, 2, 1, 1, 0, 1, 1, 0]
```

Both heuristics chose the optimal path cost with the same path and the same number of expanded vertices.

4. יבש (1נק׳): אם נגדיר שאפסילון שווה לאינסוף איך תהיה ההתנהגות של האלגוריתם עם סביבת כדורי הדרקון.

We will get that whenever we are looking to fill in the focal_queue with nodes, all of the nodes from open_list are chosen, meaning that focal_queue = open_queue and the algorithm will behave like WeightedAStar.

שאלה 12 – Benchmarking (2 נק׳):

בשאלה זאת נשווה בין אלגוריתמי חיפוש שונים על בעיות שונות. הריצו את החלק הרלוונטי במחברת ותיראו שנוצר קובץ csv. (ניתן לפתוח עם Excel).

1. **רטוב**: הריצו את החלק הרלוונטי במחברת ותיראו שנוצר קובץ csv. (ניתן לפתוח עם Excel).

WA* (0.9) expanded	WA* (0.9) cost	WA* (0.7) expanded	WA* (0.7) cost	WA* (0.5) expanded	WA* (0.5) cost	BFS-G expanded	BFS-G cos	map
240	118	200	118	224	118	445	140	map12x12
707	195	604	178	651	178	858	215	map15x15
1002	188	587	188	684	188	1045	203	map20x20

2. יבש (2 נק׳): הסבירו את התוצאות. האם הן תואמות לציפיות שלכם? האם התוצאות היו משתנות עם יוריסטיקה יותר מיודעת? נתחו והסבירו את התוצאות במונחים של מספר פיתוחים, מסלול מוחזר ומחיר הפתרון. שימו לב שבסעיף זה אין תשובה נכונה או לא נכונה אבל נדרש ממכם לספק הסבר מפורט ומבוסס.

Weighted A*:

As we can see for the 12x12 map, the returned cost for h=0.5,0.7,0.9 is the same which is 118. But for h=0.7 the number of expanded nodes is the most optimal out of the 3 weights. The same can be noticed for the 15x15 and 20x20 maps, except that for the 15x15 map h=0.9 returned a more costly path(195) in comparison with the other weights that returned 178 path cost. This can suggest that the best weight for the WeightedA* is closer to 0.7 rather than to the other weights since it returns the minimal cost and expanded nodes out of the 3 weights.

BFS-G:

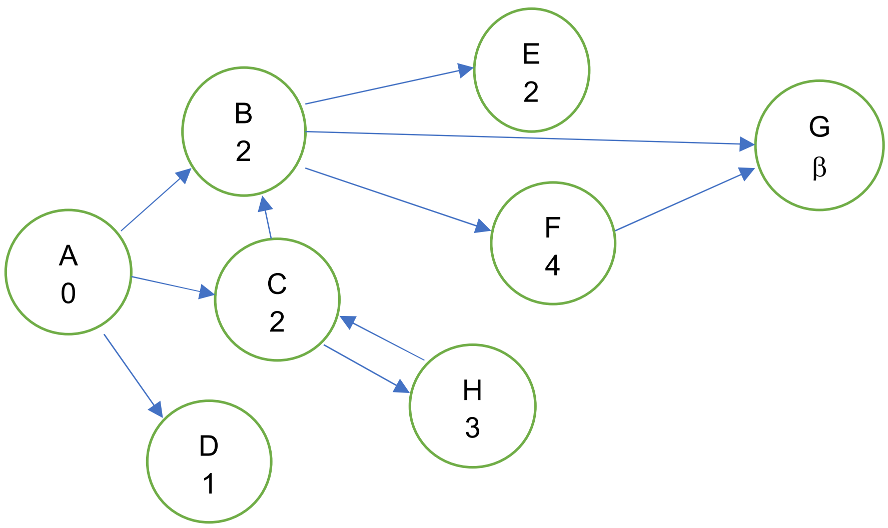
Obviously, the bigger the map is, the more costly the path to the goal might be, and so are the number of expanded nodes since there is much more to inspect and check when the map is bigger.

BFS-G vs WA*:

In the 3 maps that we tested, we can obviously see that the WA* is much more superior to the BFS-G which matches our expectations since in BFS-G we don't compare between costs and we don’t try to optimize our path. We can notice that in the returned cost which is higher than all the 3 WA* weights for all the 3 maps and so is the number of expanded nodes. and as I said, this matches our expectations.

שאלה 13 – Local Search (5 נק׳):

בהינתן מרחב המצבים הבא, כאשר a הינו המצב ההתחלתי, $U: S \to \mathbb{R}^+$ הינה פונקציית ערך והערך עבור כל מצב מצוין בצומת. המטרה שלנו היא למצוא מצב שממקסם את ערך U.



נשתמש באלגוריתם <u>Stochastic Hill Climbing</u>.

כמו כן ידוע כי $\boldsymbol{\beta > 3}$.

1. יבש (1 נק׳): מה ההסתברויות למעבר מהמצב ההתחלתי לכל אחד מהמצבים b, c, d. רשמו את $p(d|a).p(b|a), p(c|a)$

$$p(d|a) = \frac{\Delta(d,a)}{\Delta(d,a) + \Delta(c,a) + \Delta(b,a)} = \frac{1}{5}$$
$$p(b|a) = \frac{\Delta(b,a)}{\Delta(d,a) + \Delta(c,a) + \Delta(b,a)} = \frac{2}{5}$$
$$p(c|a) = \frac{\Delta(c,a)}{\Delta(d,a) + \Delta(c,a) + \Delta(b,a)} = \frac{2}{5}$$

2. יבש (1 נק׳): מה הוא מספר הצעדים המקסימלי שהאלגוריתם יכול לבצע? צעד מוגדר כמעבר בין מצבים.

<u>4, A->C->B->G</u>

3. יבש (1 נק׳): בהינתן שבצעד הראשון האלגוריתם עבר למצב c. האם האלגוריתם יתכנס למקסימום הגלובלי?

Yes.

4. יבש (1 נק׳): מה ההסתברות שהאלגוריתם יתכנס לפתרון לא אופטימלי (שאינו מקסימום גלובלי)?
יש שני מקרים :

$1 - p(c|a) = \frac{3}{5}$

5. יבש (1 נק׳): עבור אילו ערכים של β ההסתברות להגיע מהמצב ההתחלתי למקסימום הגלובלי תוך בדיוק 3 צעדים גדול מ $\frac{1}{5}$?

$4 < \beta < 7$

הוראות הגשה:

עליכם להגיש קובץ יחד בשם AI1_<id1>_<id2>.zip (בלי הסוגריים המשולשים) המכיל:

1. קובץ בשם AI1_<id1>_<id2>.pdf שמכיל את התשובות לחלק היבש.
2. קובץ בשם Algorithms.py המכיל את המימוש לאלגוריתמי החיפוש.